RUNOFF

A Program for the Preparation of Documents

Larry Barnes

System Implemented By

Larry Barnes and Larry Barusch

Document No. R-37

Issued February 18, 1969
Revised March 14, 1969

Contract No. SD-185

Office of the Secretary of Defense

Advanced Research Projects Agency

Washington, D. C. 20325

1.0 Introduction

RUNOFF is a document preparation program inspired by the program of the same name written by J. Saltzer at Project MAC. ~~This~~ version has been improved to provide more extensive control over the format of the resulting document. Control over the output is exercised by the user in two ways:

(1) By the use of special control characters inserted in text. ~~See Section~~ 1.1.

(2) By the use of command lines inserted in the text. These lines begin with dot. See Section 2.

Text for input may be prepared using the QED text editor. To use RUNOFF type "RUNOFF" to the Time-Sharing Executive. RUNOFF will respond with "LAYOUT FOR: " and the user should type T for TELETYPE, P for PRINTER, or O for OTHER, followed by a "." or ",". (This command is used to determine the physical limits of the printing device.) Then the program types "INPUT FROM: ", at which point the user should type the input file name, terminated with either "." or ",". RUNOFF then transforms the input file into an intermediate form, which is saved on the file /$RUNOFF TEXT$/. While RUNOFF is processing in pass I, any erroneous lines will be designated by the message "ERROR IN LINE n", where n is the QED line number of the line on which the error was detected. In certain cases the actual error is in a preceeding line.

If the input file name was terminated with ",", the program will type "MORE? " after processing the input file. The user should then type either Y for YES or N for NO followed by ".". If the answer is YES, RUNOFF will ask for the name of another input file.

At the start of pass II, the program will type "OUTPUT TO: ", and the user should give a scratch file name, terminated with either "." or ",". If the file name is terminated with ",", RUNOFF will type "INTERACTIVE HYPHENATION? ", and the user should respond with either Y for YES, or N for NO, followed by a ".". (See Section 2.2 for a description of the interactive hyphenation process.) Errors detected in pass II are marked with '*' in the output. RUNOFF will return to the Executive upon completion of the second pass. The user can then use the PRINT program to print the formatted file (see document R-36).

1.1 Input Text Format and Format Control Characters

Text input to RUNOFF consists of a symbolic file containing both the document to be printed and the control information for formatting. A physical line is a string of characters ending with carriage return and line feed. All physical lines which do not begin with "." are considered lines of the document to be output. Initially, RUNOFF is set to "FILL" mode. All physical lines are collected into a single, logical, line up to the occurence of a line break. The logical line is then broken into as many physical output lines as are needed to fit within the line margins. These lines will then normally be adjusted by inserting extra spaces in mid-line so that the end of the line is on the right margin. The end of a logical line, or line break, occurs when one of the following conditions is met:

(1) There are one or more blank lines between two lines of text. (Extra blank lines in the text are ignored in fill-mode. The user must use the appropriate command to produce spaces.)

(2) The next line begins a new paragraph. (RUNOFF recognizes paragraphs by seeing one or more blanks at the beginning of a line of text.)

(3) There is a logical line break. This is produced by the command BREAK and certain other command lines. (See the summary for a full listing.)

Any character in an actual text line is interpreted as text except for the escape, shift, or tab character. The escape character is initially "¶" but may be changed by command (see Section 2.8); the initial shift character is "/", and tab is initialized to "≠" (see Section 2.8). The escape character and the immediately following one have special meanings as defined below:

¶A causes all subsequent upper case characters in the text to be printed as lower case characters. This changes ASCII characters 40-137 to 100-177 octal.

¶L converts upper case letters to lower case. ASCII characters 41-72 become 101-132 octal. This is the initial mode.

↑U	Upper case-Do not convert to lower case and ignore "↑S" convention.
↑S	Shift-Do not convert the next upper case character as above. The shift character "/" has the same effect as "↑S".
↑D	converts the subsequent character to opposite case. For example, "↑D<" converts the "<" to ">" in normal mode.
↑C	capitalizes the next word up to a space, carriage return, punctuation mark, or ↑E.
↑B	Backspace-Overprint the preceeding character with the next character.
↑I	underlines the next word up to a space, carriage return, punctuation mark, or ↑E.
↑-	Underlines all characters up to a carriage return, or ↑E.
↑E	is used to force the end of underlining and capitalization before the normal boundary occurs.
↑T	denotes the end of a field for formatting; the tab character "\" may also be used.
↑Y	causes the current date to be output.
↑↑	Print an up arrow.
↑/	Print a slash
↑\	Print a backslash. These control characters are modified in the obvious manner if the escape, shift, or tab characters are changed.

Any uninterpretable character combination will be deleted and an error message will be printed.

BLANK PAGE

2.0 General Information about Commands

A command line begins with ".". All letters are converted to upper case in recognizing the names of commands, macros, or formats. Following the dot RUNOFF expects a command name or an abbreviation, followed by the parameters for that command, followed optionally by a comment. Lines beginning with dot but having an unrecognizable format are treated as errors. No lines beginning with dot are printed unless the preceeding line was .LITERAL. All commands are described below. Abbreviations for command names are normally based on the first two letters of a one word command or the first letter of the first two words of a multi-word command. Commands which should cause a logical line break do. Information on abbreviations and whether commands cause line breaks will be found in the summary at the end of the manual. In a command line RUNOFF will consider multiple blanks as a single blank, if a blank character is legal.

2.1 General Actions

.fill
.nofill
.format <name>

These commands determine which line-processing routine should be used. ".fill" causes RUNOFF to either lengthen short lines by filling with words from the following line or shorten long lines by breaking between words. Filling will not take place across a line break. In nofill-mode each input line produces one output line; further blank lines _are_ output in this mode. ".format" causes subsequent text to be output under the control of the specified format (see below). Each following logical line will be fit into the format until a ".fill" or ".nofill" command is encountered.

.adjust
.nojust

The adjust command causes all lines processed in fill-mode, except the last one before a line break, to be right justified. This is the initial state of RUNOFF.

.define format <name> <pos> <field definition>
.end format

These commands define a format for use in producing tables, etc. The <name> identifies the format; the position <pos> may be one of LEFT, RIGHT, or CENTER, and determines the overall position of the format with respect to the margins. Each <field definition> has the form:

```
<type>(<letter> ... <letter>)
```

where the <type> is one of L for left, R for right, C for center, F for fill, or J for justify. The first three types define fixed fields; the text to be formatted must fit within the allotted space. The latter types define variable fields; the text will be handled as in normal fill mode processing.

A picture showing the manner in which text should be output follows the ".define format" command; following the picture should be an .end format command. The following lines give an example:

```
.define format SUMMARY l(A) f(C) c(B)
AAAA  CCCCCCCCCCCCCCCCCCCCCCCCC  BBBBBBB
      CCCCCCCCCCCCCCCCCCCCCCCCC
.end format
```

The first field of text is left-justified; the second is centered; the third is subjected to fill-mode processing without justification. After the first line of output is generated using this format, all subsequent lines are produced using the last picture line. (Strictly speaking the third line is unnecessary.)

Text for formatted processing consists of a logical line (or paragraph). Each field except the last must be separated by a tab ("/" or "¶T"). The first field of text is A; the second, B etc. Typical input for our example might be:

```
1A/YES//THIS IS SOME TEXT
TO BE FILLED.
```

The characters in the picture lines are interpreted as follows. Contiguous sequences of letters determine the field positions; non-alphabetic characters are output literally. (Note: "q.qq" will not work; put the "."

in the text.) A sequence of characters written between double quotes is considered literal text. The double quotes are not output, and there is no way to use double quote as a literal.

2.2 Hyphenation Processing

When interactive hyphenation is specified, RUNOFF will type "HYPHENATE: <word>" when it finds a word which should be hyphenated and which is not in the glossary. The user should then type the word with hyphens, and end the line with carriage return. Control A and control Q may be used to correct typing errors. If only carriage return is typed, RUNOFF will not try to hyphenate the word. Interactive hyphenation may be turned off by ending a word with control D instead of carriage return.

At the end of processing, RUNOFF writes the glossary on the file '/$GLOSSARY$/'. If RUNOFF aborts during processing, "CONTINUE RUNOFF." may be used to dump the glossary.

There are four conditions which must be met before hyphenation takes place:

(1) RUNOFF must be in hyphenation mode.

(2) The text at the end of the line must consist of a non-alphanumeric prefix, followed by an alphabetic string (which may also include "-", "/", and "'" characters), followed by a non-alphanumeric suffix. The prefix and suffix may be empty.

(3) There must be at least two characters before the hyphenation point and at least three characters after it.

(4) The number of spaces to be inserted per word on the line must be greater than the .hyphenation break parameter (see below).

If the word meets these conditions, then it will be broken at a "-" or "/" (if any), or at the right-most hyphen which fits the line if it is in the glossary.

.hyphenate
.nohyphen
: In hyphenation mode RUNOFF will try to find a word in the glossary which is the same (except for the endings -S, -ES, -ED, and -F) as the word at the end of the line of text. RUNOFF is initially in hyphen mode but a null glossary produces nearly the same effect as NOHYPHEN mode.

.glossary W
: This command inserts words into the glossary for use in hyphenation. Each word should have the form "hy-phen-ate" and be separated by spaces. (The double quotes should not be present.)

.hyphenation break N
: This command sets the parameter which determines the allowable number of spaces to be inserted in a line before RUNOFF tries to hyphenate the last word. Each space counts ten points. If more than N points per word would have to be inserted, then hyphenation will be attempted. The initial setting of this parameter is 5 (one-half space per word).

2.3 Margin Controls

There are two types of margins involved in RUNOFF:

(1) The physical margins. These are determined by the nature of the printing device. The margins outline the area where it is physically possible to print characters, and are usually set with the LAYOUT option (see Section 1.0).

(2) The logical margins. These can be set by the user as he wishes. (Limits are imposed by the physical margins.) They are initialized for standard 8.5" by 11" printing.

Commands concerning vertical and horizontal margins are:

.page layout TM, BM, TOL
: This sets the vertical logical margins and vertical tolerance. Parameters are top margin, bottom margin and tolerance. The tolerance is used to determine where to break between pages on page overflows. If

there is a line break within TOL lines of the bottom, RUNOFF will break the page there; otherwise it will fill the page completely.

.line layout LM, RM, NC, CS
This sets the logical left and right margins, the number of columns, and the number of spaces to insert between columns. These margins are used for the page headings. To adjust the relative text position, use the subsequent commands.

.reduce margin LM, RM
.expand margin LM, RM
.end reduction
These commands enable the user to indent a certain portion of his text using the first command, or "undent" his text using the second command. In either case the original margins are restored by the third command. The use of several ".reduce margin" commands before the corresponding ".end reduction" commands succesively indents the text more, and more. Thus these commands are like brackets (<u>ie</u> recursive). LM is added to the left logical margin and RM is subtracted from the right logical margin in the first command. Just the opposite is done on the second command. Negative numbers are permitted. These commands do not affect the position of page headings.

.layout PLM, PRM, PTM, PBM, LL, LO
This command defines the physical margins in the following complex manner. (It should only be used for non-standard devices; normally this command should <u>not</u> be necessary.) The parameters are the physical left margin (in spaces), the physical right margin, the physical top line, the physical bottom line, the line length, and line origin. The first four parameters define the physical limits of the printing device. The final two parameters define the length of the logical line and its origin with respect to the left edge of the paper. Printing starts at column LO + LM, and ends at LO + RM, where LM and RM are the <u>logical</u> margins established by ".line layout". When using the "facing" feature (see ".paging mode"), the logical left margin is LL - RM on even pages, and the right margin is LL - LM. The parameters for the layout command must satisfy:

```
    min(LO + LL - PLM, PRM - LO) >
                      max(PLM - LO, LO + LL - PRM),
            LL > 25, and PBM - PTM > 6.
```

This command sets LM to 15, RM to LL - 10, TM to PTM, and BM to PBM - 6. (These margin settings produce the standard 1.5 inch left, and 1 inch right, top, and bottom margins.)

Initially RUNOFF sets the margins for teletype output to:

```
.layout 6, 89, 6, 66, 85, 0
.line layout 15, 75
.page layout 6, 60, 4
```

The printer layouts is:

```
.layout 5, 137, 6, 66, 85, 15
.line layout 15, 75
.page layout 6, 60, 4
```

The logical margins must satisfy:

$$\min(LL, PRM - LO, LO + LL - PLM) \geq RM > LM \geq \max(0, PLM - LO, LO + LL - PRM),$$
$$PBM \geq BM > TM \geq PTM, \text{ and } BM - TM > TOL.$$

2.4 Paragraph Formatting

.paragraph spacing N
This specifies how many lines are to be inserted beteen paragraphs. Initial setting = 1.

.paragraph indentation N
This specifies how many additional spaces to insert at the beginning of a paragraph. Initial setting = 5.

.paragraph undentation N
This command is the same as ".paragraph indentation -N". That is, N fewer spaces are inserted at the beginning of the paragraph.

2.5 Special Line Justification and Control

These commands pertain to the next logical line. The end of the line should be designated with a break.

.center
Center the next line.

.indent N
    Indent the next line N spaces. If N is not provided, 5 is assumed.

.undent N
    Start the next line N spaces to the left of the normal margin. This command is the same as ".indent -N".

.margin
    Justify the next line against the right hand margin.

2.6 Heading and Paging

.header XXXXXXX
    RUNOFF accepts a heading to go on the first line of each page. The heading string is assumed to start at the first non-blank character after the control word and end at carriage return.

.heading mode <param>
    <param> determines the postion of the heading on the line. <param> may be any of the following.

    CENTER
        The header will be centered on the line.

    MARGIN
        The header will be adjusted against the right margin.

    FACING
        on even numbered pages the header is adjusted against the right margin. On odd pages it is adjusted against the left margin.

    OPPOSED
        the header will be adjusted against the opposite margin from the page number. This is the initial mode.

.paging mode <param>
    This command determines the placing of the page number. All parameters are optional. <param> may be any one or more of the following commands. In case of conflict the latest command wins.

CENTER
  The page numbers are centered between the logical margins.

MARGIN
  The page number is adjusted against the right margin.

FACING
  On even numbered pages the number will be adjusted against the right margin. On odd numbered pages the number will be adjusted against the left margin.

TOP
  Page numbers are placed on the first line.

BOTTOM
  Page numbers are placed on the last line.

OFF
  Printing page numbers is discontinued.

PREFIX "<string>"
SECTION "<string>"
SUFFIX "<string>"
  The strings of characters between quotation marks are used to form the page string, which has the form:

```
<prefix><section><page number><suffix>.
```

  Any or all of these strings may be null. The section string is considered to be part of the page number for purposes of indexing.

Initial mode is:

```
.paging mode TOP MARGIN PREFIX "Page"
.paging mode SECTION "" SUFFIX ""
```

If neither page number nor heading is used, the text will start on the first logical line. Otherwise it will start on the fourth logical line. If the page number is at the bottom, text will end on the fourth line from the bottom. If the paging and heading mode conflict, the page string overwrites the heading.

.odd page
    This control word causes the current page to be printed out and the next page to be started with the next higher odd number.

.page N
    If N is present, insert a page break and start numbering the next page with N. Otherwise, turn the paging mode on and do not insert a page break.

.eject N
    Insert a page break if either there are fewer than N lines left on the page or N is not present.

2.7 Lines and Spacing

.single space
    Single space all lines within paragraphs. This is the initial state.

.double space
    Double space all lines within paragraphs.

.space N
    Output N line spaces. If N is not provided, 1 is assumed. In case of page overflow all remaining blank lines to be output are deleted.

.figure spacing N
    This command is equivalent to ".eject N" followed by ".space N". These commands provide the only means of creating blank lines.

.break
    The lines before and after this command will not be run together in fill mode. A simpler way to get a line break is to insert one or more blank lines in the text.

.begin group
.end group
    The output lines enclosed between these two commands are forced to lie on a page. Thus this command acts in a manner similar to ".eject N", where N has the 'right' value.

2.8 Miscellaneous

.underline
    The following line is underlined.

.literal
    The next line is taken as part of text whether or not it begins with dot.

.escape <char>
.shift <char>
.tab character <char>
    The given character becomes the escape, shift, or tab character. The parameter for the .shift and .tab character commands may be null, if no shift or tab character is desired.

.define command <name>
.end command
.call <name>
    These commands give the user the opportunity to combine text and control lines to form his own commands. All text and command lines between the first and second commands is stored away under NAME. When the third command is executed, the stored string is read and the commands within the string are executed. Recursion is not permitted.

.index <phrase>, <phrase>
    RUNOFF saves the first phrase in the main index table and the second phrase (if any) in a sub-index table associated with the first phrase.

    The index is formatted and output after the last page of text. Two built-in but redefinable formats, RINDEX and SINDEX, are used to format the index as shown in the following example.

```
          Algorithms, 40, 78,    < uses RINDEX
            analysis of, 27,     < uses SINDEX
```

    The following lines give the initial definitions for the indexing formats.

```
        .define format RINDEX  f(A)
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAA
            AAAAAAAAAAAAAAAAAAAAAAAAA
        .end format
```

```
.define format SINDEX f(A)
  AAAAAAAAAAAAAAAAAAAAAAAAA
     AAAAAAAAAAAAAAAAAAAAAAA
.end format
```

In order to get an index output in two columns, ".line layout 15, 75, 2, 4" should be the last line of the input.

In the summary the following conventions are used for parameters.

Symbol	Meaning
N	a number
C	a character
S	a string
W	a word (or name)
P	a general parameter

Any symbol may be followed by *, indicating that its use is optional as that parameter of the command. If an optional parameter is left out, the effect is to not change the state of RUNOFF, except where stated otherwise above. The use of + following a break type indicates that the actual break is dependent on the value or presence of the parameter.

Abbreviation	Command	Type of Automatic Break	Section
.ad	.adjust	line	2.1
.bg	.begin group	none	2.7
.br	.break	line	2.7
.cl	.call W	none	2.8
.ce	.center	line	2.5
.dc	.define command W	none	2.8
.df	.define format W P ... P	none	2.1
.ds	.double space	line	2.7
.ej	.eject N*	page+	2.6
.ec	.end command	none	2.8
.ef	.end format	none	2.1
.eg	.end group	none	2.7
.er	.end reduction	line	2.3
.es	.escape C	none	2.8
.em	.expand margin N*, N*	line	2.3
.fs	.figure spacing N	line	2.7
.fi	.fill	line	2.1
.fm	.format W	line	2.1
.gl	.glossary W ... W	none	2.2
.he	.header S	none	2.6
.hm	.heading mode P	none	2.6
.hy	.hyphenate	none	2.2
.hb	.hyphenation break N	none	2.2
.in	.indent N*	line	2.5
.ix	.index S, S*	none	2.9
.la	.layout N*, N*, N*, N*, N*, N*	page	2.3
.ll	.line layout N*, N*, N*, N*	column	2.3
.li	.literal	none	2.8
.ma	.margin	line	2.5

Abbreviation	Command	Type of Automatic Break	Section
.nf	.nofill	line	2.1
.nh	.nohyphen	none	2.2
.nj	.nojust	line	2.1
.op	.odd page	page	2.6
.pa	.page N*	page+	2.6
.pl	.page layout N*, N*, N*	page	2.3
.pm	.paging mode P ... P	none	2.6
.pi	.paragraph indentation N	none	2.4
.ps	.paragraph spacing N	none	2.4
.pu	.paragraph undentation N	none	2.4
.rm	.reduce margin N*, N*	line	2.3
.sh	.shift C*	none	2.8
.ss	.single space	line	2.7
.sp	.space N*	line	2.7
.tc	.tab character C*	none	2.8
.un	.undent N*	line	2.5
.ul	.underline	line	2.8